



caBIG

*cancer Biomedical
Informatics Grid*



caBIG Compatibility Guidelines

June 14, 2004

caBIG Compatibility Guidelines

2

► What are the caBIG Compatibility Guidelines ?

- Provide Guidelines on caBIG Compatibility for:
 - caBIG participants
 - Interested stakeholders on caBIG compatibility
- Set of definitions that caBIG participants can use to measure the maturity level of potential caBIG applications.

caBIG Compatibility Guidelines

3

▶ How do they differ from caBIG Principles

- The caBIG Principles do not provide the guidelines for Applications to become caBIG compliant
- The caBIG Principles are the core values of the program, and apply to projects which are funded by NCICB
- Unfunded or vendor projects can be caBIG compliant (by following the Compatibility Guidelines) without following all the caBIG Principles
- Both are important parts of a functioning, community-wide caBIG

caBIG Compatibility Guidelines

4

► When will the Guidelines be complete?

- On-going effort
- Evolving, “Living Document”
- Driven by input and participation of the caBIG Community

caBIG Compatibility Guidelines

5

► How are the Guidelines to be used?

- “Remediation” of legacy Applications
- “Evaluation” of existing Applications
- “Creation” of new Applications

caBIG Compatibility Guidelines

6

► What is the structure of the Guidelines?

- Levels of Maturity:
 - “Legacy”
 - “Bronze”
 - “Silver”
 - “Gold”

caBIG Compatibility Guidelines

► Grading - Matrix

Maturity Model	LEGACY	BRONZE	SILVER	GOLD
Interface Integration	<ul style="list-style-type: none"> - No Programming interfaces to the system are available. Only local data files in a custom format can be read - Some ad hoc data transfer mechanism such as FTP 	<ul style="list-style-type: none"> - Provide baseline* programmatic access to data. Data can be read from remote electronic sources or from commonly used file formats. Data can be pushed out to from applications to other external data sources. 	<ul style="list-style-type: none"> -- Well-described API's that provide access to data objects. -- System architecture separated into tiers and interoperable components -- Data read in from standards-based electronic sources that support standard or commonly used interchange formats -- Documented component description of the underlying data structures that are accessible -- Standard messaging systems where appropriate 	<ul style="list-style-type: none"> - All features of Silver, plus: - Interoperable with data grid architecture to be defined by caBIG - Fully componentized provide access to individual resources in the form of grid services
Vocabularies / Terminologies & Ontologies	<ul style="list-style-type: none"> - Free text used throughout for data collection 	<ul style="list-style-type: none"> - Use of publicly accessible standardized controlled vocabularies as well as local terminologies 	<ul style="list-style-type: none"> - Messages need to be based on a common messaging model (such as HL7) 	<ul style="list-style-type: none"> - All features of Silver, plus: - Fully compliant with caBIG recommended standards for vocabulary terminology services and content sources - Common Data Elements should be built using controlled terminology
Data Elements	<ul style="list-style-type: none"> - No Structured metadata is recorded 	<ul style="list-style-type: none"> - Some type of metadata describing the information in the system is used for data collection and external reporting. Metadata is retrieved from external repository shared by multiple applications. 	<ul style="list-style-type: none"> - Use common standard electronic representation for CDE's such as ISO 11179 or comparable standard - CDEs are harmonized and re-used from across the Domain Workspace 	<ul style="list-style-type: none"> - All features of Silver, plus: - Programmatic access to all metadata, including data class descriptions, site and source information, and any other caBIG-defined metadata requirements and use information models - Use the caBIG standard or electronic representation of metadata and Common Data Elements
Information Models	<ul style="list-style-type: none"> - No particular information model is used to represent data 	<ul style="list-style-type: none"> - Some type of diagrammatic model describing the data relationship is available in electronic format 	<ul style="list-style-type: none"> - Information models defined in a standard modeling language such as UML 	<ul style="list-style-type: none"> - All features of Silver, plus: - Information models are harmonized with others across the caBIG Domain Workspace

caBIG Compatibility Guidelines

8

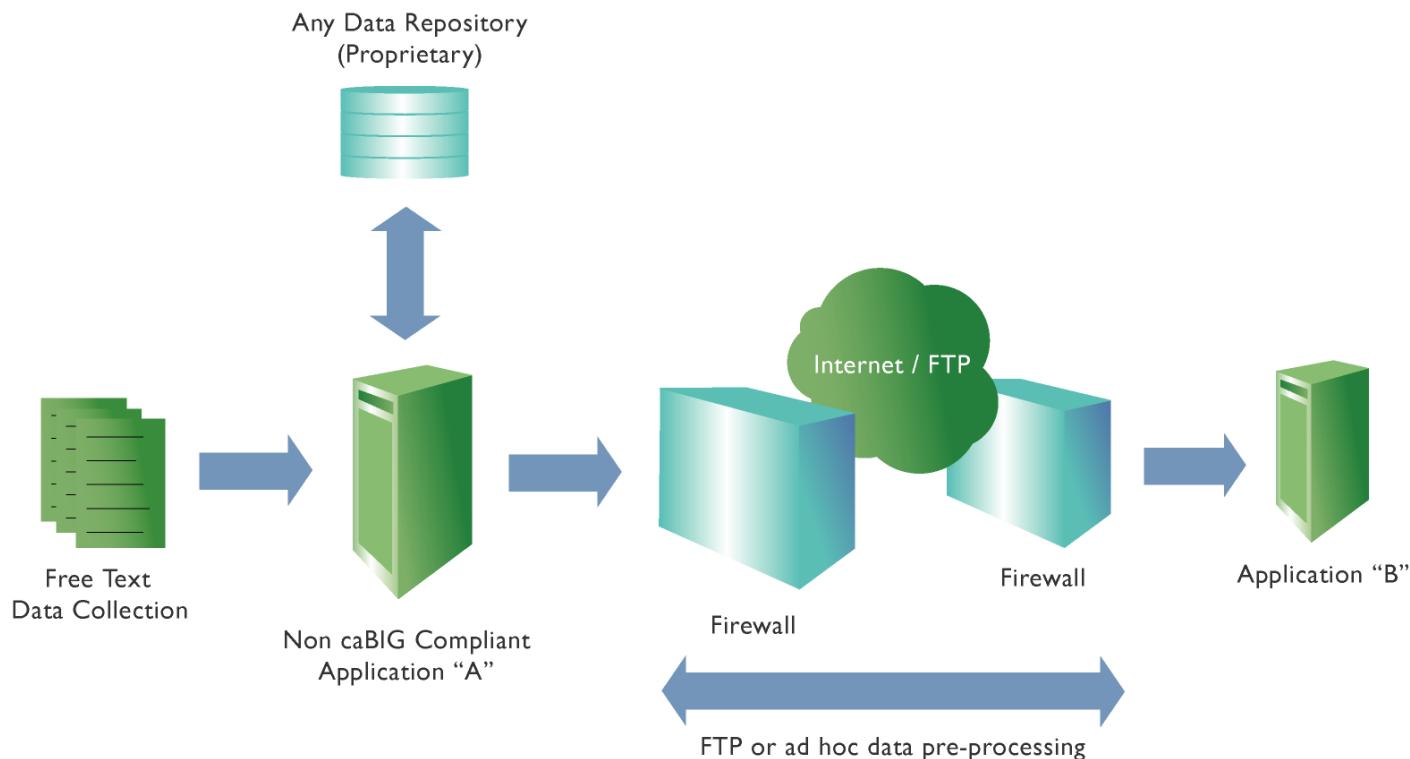
► Requirements

- Minimal requirements – “Bronze”
- New Development “Silver”
- Deprecating “Bronze” in c. 24 Months
- caBIG Community defines “Gold”

caBIG Compatibility Guidelines

9

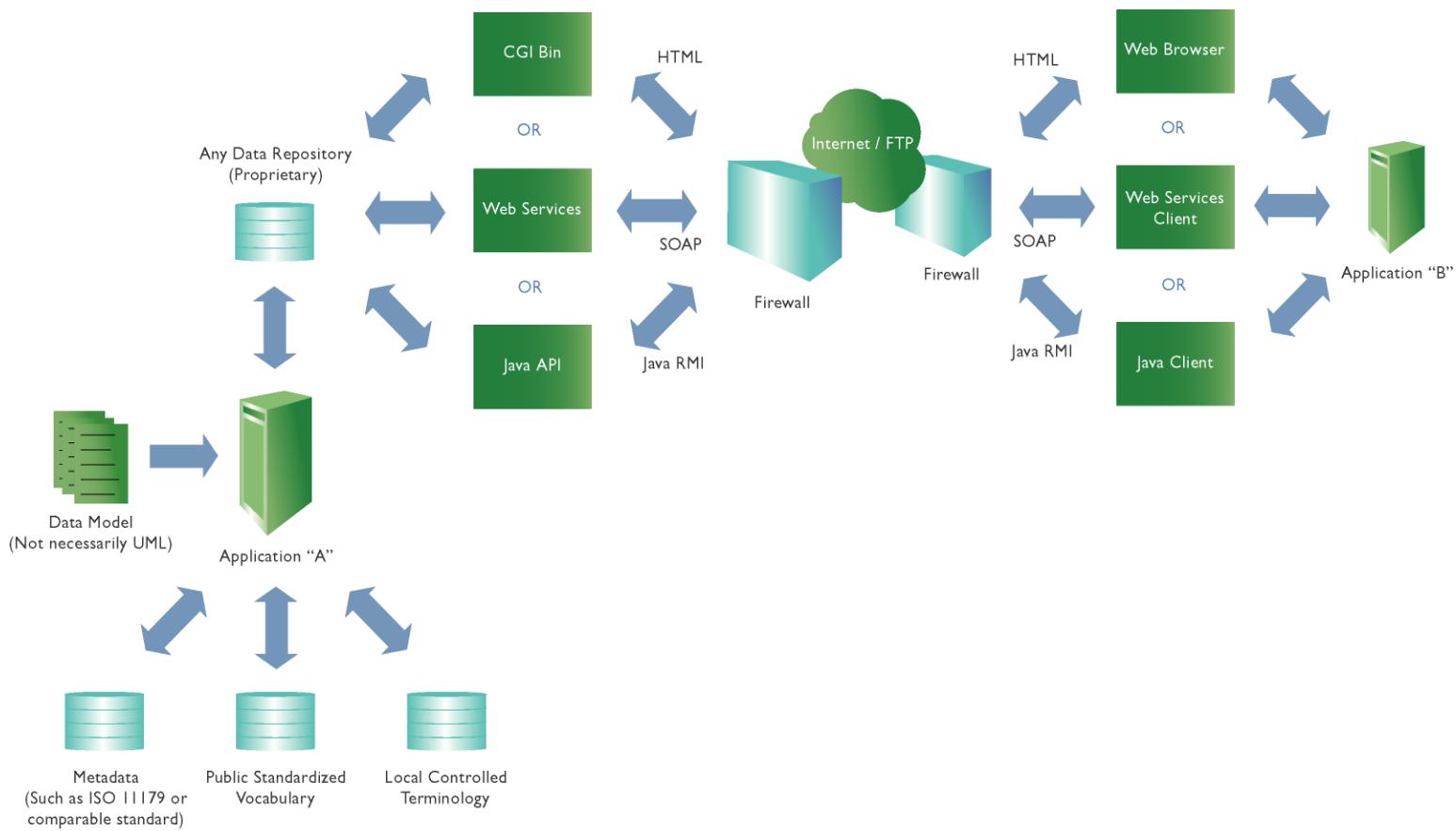
Outline of “Legacy” Level of Maturity



caBIG Compatibility Guideline

10

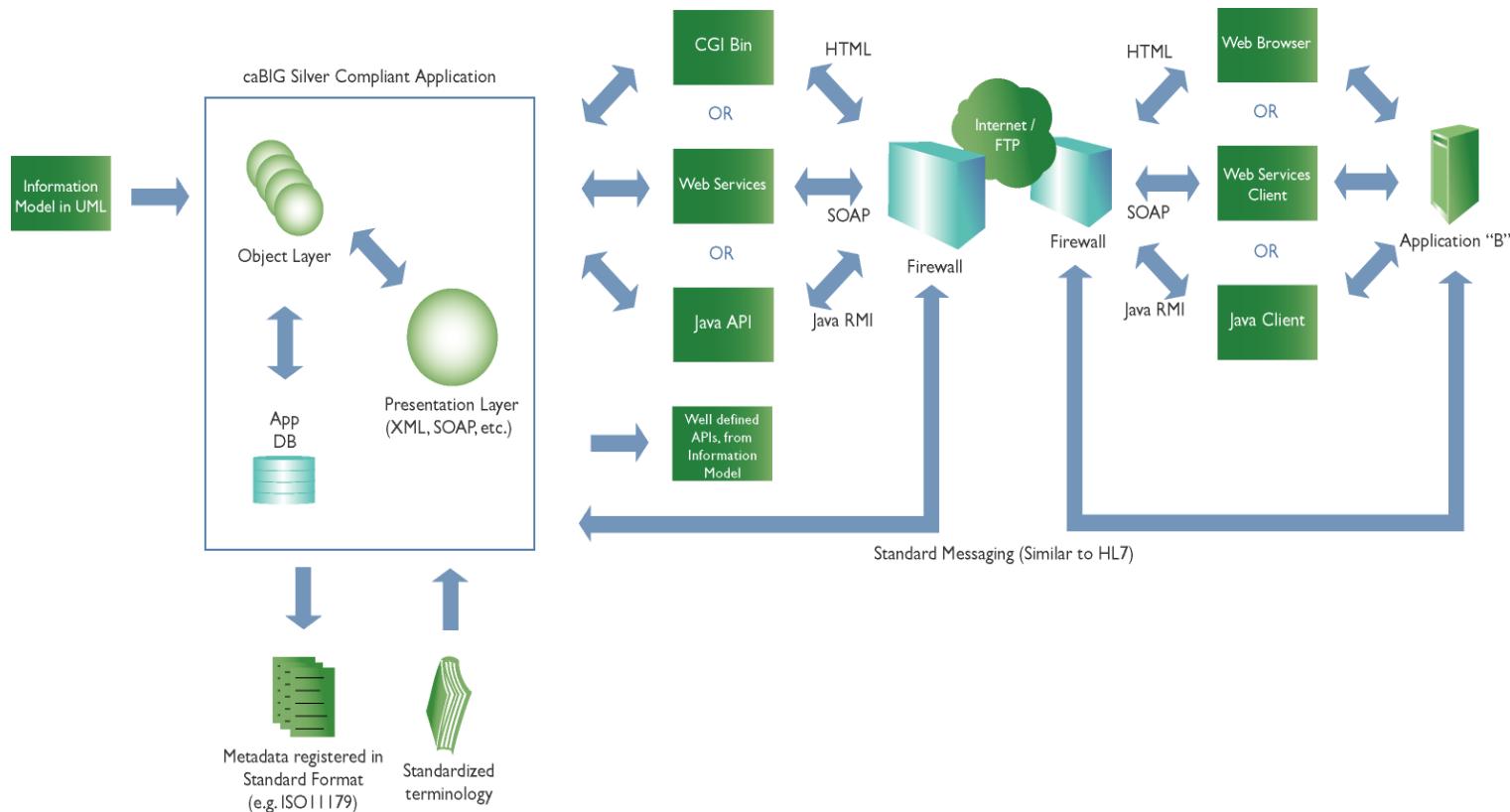
Outline of “Bronze” Level of Maturity

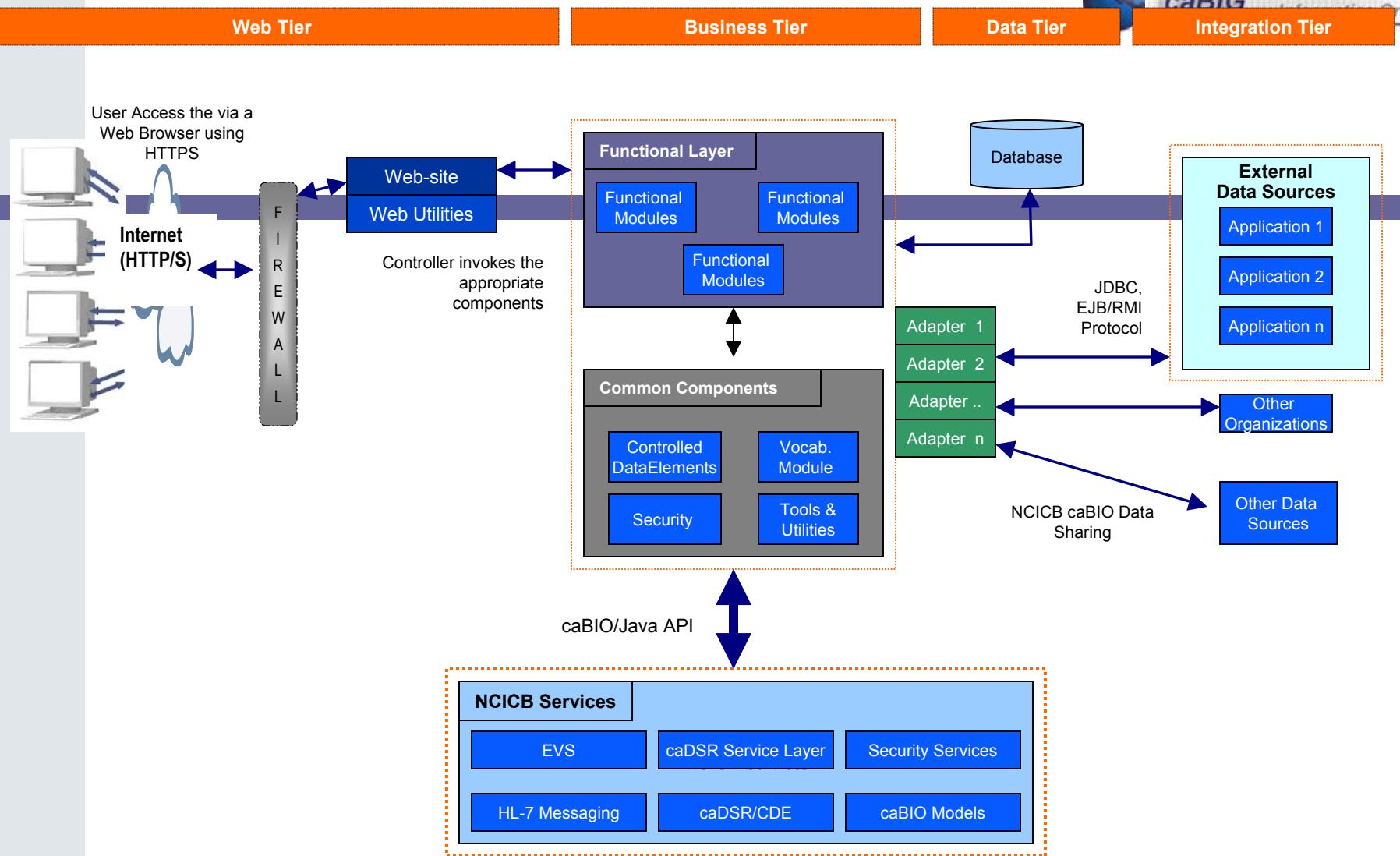


caBIG Compatibility Guideline

11

Outline of “Silver” Level of Maturity





BRONZE MATURITY LEVEL

Interface Integration

Access to data electronically by using industry standard file formats.

Examples:

JDBC – Access to external database using JDBC over TCP/IP

Web Services – Invoke application level API using SOAP

Vocabularies/Terminologies:

Usage of Public standardized controlled vocabularies.

Examples:

Manual input of Vocabulary terms when submitting or editing data

Data Elements:

Metadata describing information in the system, stored in electronic format.

Examples:

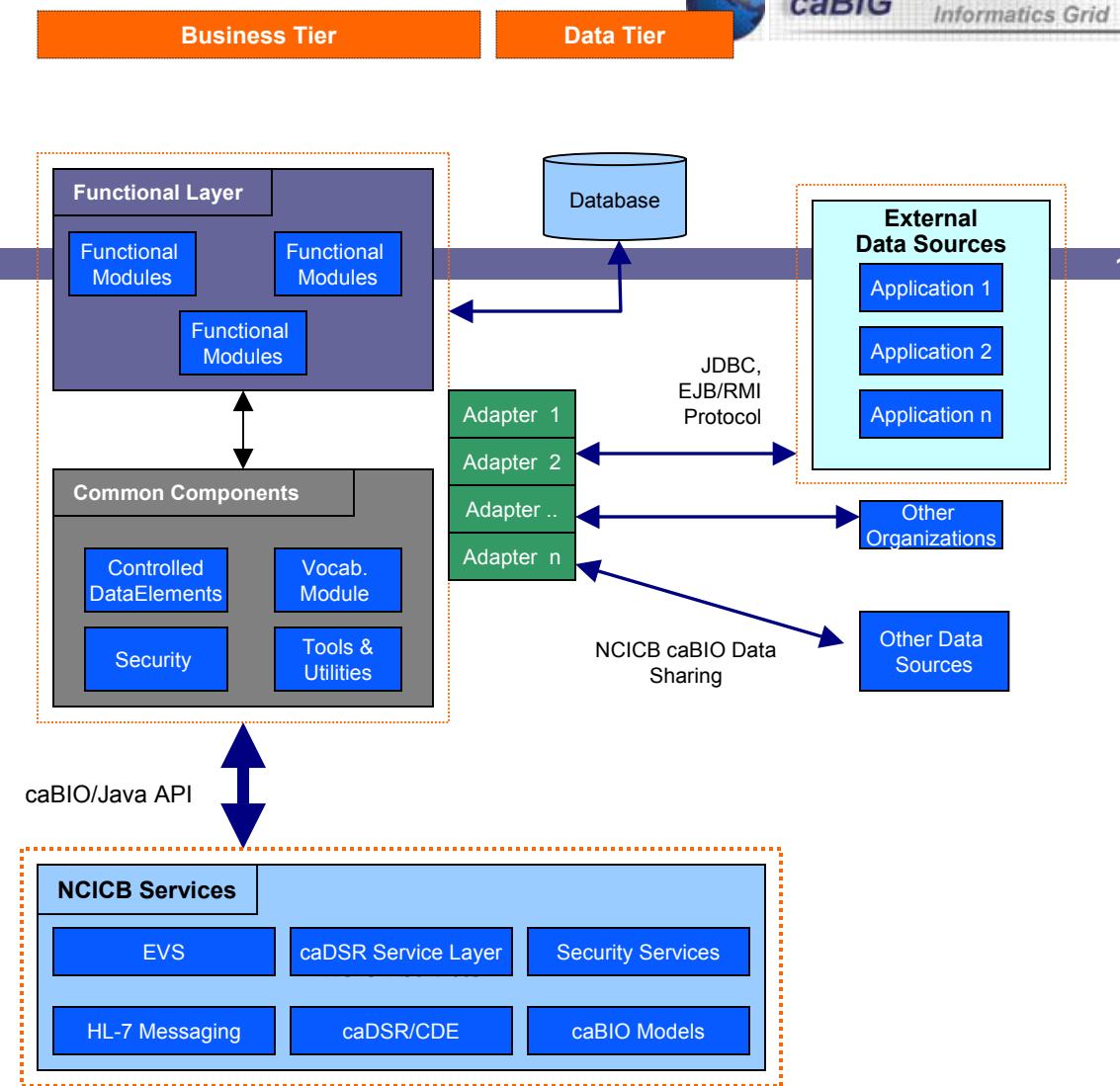
Controlled spreadsheet containing Data Elements used in the system

Information Model:

Diagrammatic model describing the data relationship available in electronic format.

Examples:

Entity Relationship Diagram of the data model





Access to data electronically by using industry standard file formats.

Examples: JDBC – Access to external database using JDBC over TCP/IP

JDBC Code Sample

*CSAERS DAO Code Sample

```
// Initialize Remote Connection using JDBC 2.0 and Prepared Statement
// create the prepared statement
try {
    prepStmt = conn.prepareStatement(userPersonalNoticeQueryString);
    // fill in the parameters from the query
    prepStmt.clearParameters();
    prepStmt.setInt(1, userVO.getMessageId().intValue());
    prepStmt.setString(2, userVO.getUserId());
    prepStmt.setString(3, userVO.getMessageText());
    prepStmt.setString(4, userVO.getCreatedBy());
    prepStmt.setTimestamp(5, new Timestamp(userVO.getCreatedDate().getTime()));
    prepStmt.setTimestamp(6, new Timestamp(userVO.getExpirationDate().getTime()));
    prepStmt.setInt(7, userVO.getMessageDefinitionId().intValue());
    prepStmt.setInt(8, userVO.getCaseId().intValue());

    // execute the stmt
resultSet = prepStmt.executeQuery();

} catch (SQLException sql_e) {
    String message = "Cannot execute personal notices list query";
    logger.error(message, sql_e);
    throw new DAOAccessException(message, sql_e);
} finally {
    // Close Result Set, Statement and Connection
}
}
```



Access to data electronically by using industry standard file formats.

Example: Web Services – Invoke application level API using SOAP e.g., caBIO SOAP API

Web Services Code Sample

*caBIO SOAP Example, using SOAP::LITE

```
$URI='urn:nci-gene-service';
$PROXY_PATH='/soap/servlet/rpcrouter';
my $argc=@ARGV;
# anonymous hash reference, passed as map to SOAP service
my $searchRec={};
# first three arguments are mandatory
if ($argc < 3) {
    die $USAGE;
} else {
    $server = shift @ARGV;
    $port = shift @ARGV;
    $method = shift @ARGV;
    $argc=@ARGV;
    while ($argc) {
        my $arg = shift @ARGV;
        if ($arg eq "-symbol") {$searchRec->{symbol} = shift @ARGV;}
        elsif ($arg eq "-genBankAccessionNumber") {$searchRec->{genBankAccessionNumber} = shift @ARGV;}
        elsif ($arg eq "-pathwayId") {$searchRec->{pathwayId} = shift @ARGV;}
        ...
        ...
        elsif ($arg eq "-returnHeavyXML") {$searchRec->{returnHeavyXML} = shift @ARGV;}
        elsif ($arg eq "-fillInObjects") {$searchRec->{fillInObjects} = shift @ARGV;}
        elsif ($arg eq "-targetId") {$searchRec->{targetId} = shift @ARGV;}
        else {
            print "\n****Please check the name of attribute****\n";
            die $USAGE;
        }
        $argc=@ARGV;
    }
}

$s = SOAP::Lite -> uri($URI) -> proxy("http://$server:$port$PROXY_PATH");

# make service request
$som=$s->$method(SOAP::Data->type(map => $searchRec));
# interpret result
if ($som->fault) {
    print "FAULT ENCOUNTERED!\nfaultcode:\t" . $som->faultcode . "\nfaultstring:\t" . $som->faultstring . "\n";
} else {
    $xmldoc = $som->result;
    print "\nMETHOD CALLED: $method\n\n" . decode_entities($xmldoc) . "\n\n";
}
```



Usage of Public standardized controlled vocabularies.

Example: EVS Lookup or Manual look-up using internal Medical Coding Libraries

EVS Code Sample

*caBIO EVS Code Samples

16

```
DescLogicConceptSearchCriteria dlcsc = new DescLogicConceptSearchCriteria();
DescLogicConcept dlc = new DescLogicConcept();
MetaThesaurusConceptSearchCriteria mtcsc = new MetaThesaurusConceptSearchCriteria();
MetaThesaurusConcept mtc = new MetaThesaurusConcept();
String vocabularyName= "NCI_Thesaurus";
String conceptname = "Gene";

//-----
//  NCI Thesaurus
//  search()
//-----

dlcsc.setLimit(10); //default is 100
dlcsc.setSearchTerm("Gen*");
dlcsc.setVocabularyName("NCI_Thesaurus");
Concept[] conceptArray = dlc.search(dlcsc);
if(conceptArray!=null) {
    for(int i=0; i<conceptArray.length; i++) {
        System.out.println(conceptArray[i].getName());
    }
}
```



Metadata describing information in the system, stored in electronic format

Examples: Controlled spreadsheet containing Data Elements used in the system

Data Element Sample

Element Name	Element Type	Minimum Length	Maximum Length	Associated Page	Data Format Validation	Required	Allowed Values
Patient_First_Name	String	1	100	REGISTRATION	N/A	✓	N/A
Patient_Last_Name	String	1	100	REGISTRATION		✓	N/A
Patient_Suffix	String	0	5	REGISTRATION		✓	Mr. Mrs. Ms.
Patient_Address_Line_1	String	1	200	REGISTRATION		✓	
Patient_Address_Line_2	String	1	200	REGISTRATION		✓	

Controlled Excel Spreadsheet to manage Data Elements associated with the System

Diagrammatic model describing the data relationship available in electronic format.

Examples: Entity Relationship Diagram of the data model

Data Element Sample

*CSAERS EVS Medical Coding ERWin Diagram

18

